**PCT** WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04N 7/10 | A1

(11) International Publication Number: WO 00/10329

(43) International Publication Date: 24 February 2000 (24.02.00)

(21) International Application Number: PCT/US99/18292

(22) International Filing Date: 11 August 1999 (11.08.99)

(30) Priority Data:
60/096,665 13 August 1998 (13.08.98) US

(71) Applicant *(for all designated States except US)*: PLAY, INC. [US/US]; 2890 Kilgore Road, Rancho Cordova, CA 95670 (US).

(72) Inventor: MONTGOMERY, J., Paul (deceased).

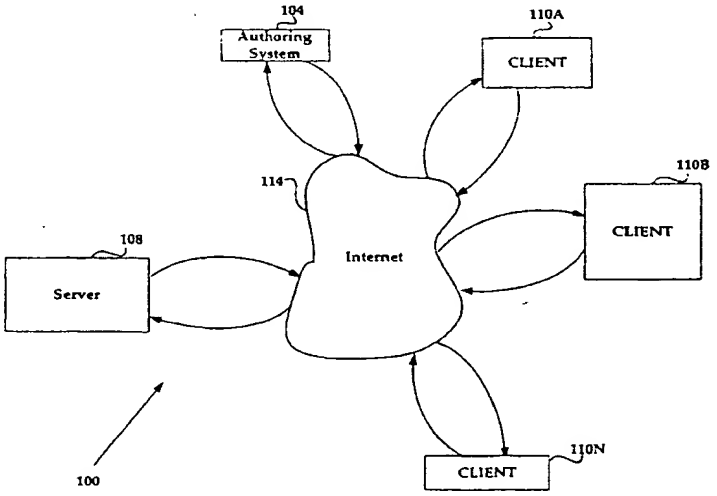
(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: HARTFORD, Stephen [US/US]; 133 Brightstone Circle, Folsom, CA 95630 (US). MOORE, Michael, Richard, Young [US/US]; 2890 Kilgore Road, Rancho Cordova, CA 95670 (US). SCHAEM, Stephan, D. [US/US]; 6635 Sylvan Road #924, Citrus Heights, CA 95610 (US). RISO, Thomas, A. [US/US]; 5432 Fleetwood Drive, Citrus Heights, CA 95621 (US). KELL, Steven, R. [US/US]; 3490 Patterson Way, El Dorado Hills, CA 95762 (US).

(74) Agent: NGO, Tuan, V.; Carr & Ferrell LLP, Suite 200, 2225 East Bayshore Road, Palo Alto, CA 94303 (US).

(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

**Published**
*With international search report.*

(54) Title: CLIENT–SIDE DIGITAL TELEVISION AUTHORING SYSTEM




(57) Abstract

A system and method for providing a client–side (110A, 110B, 110N) multimedia program enables delaying the actual final production of the video until a user begins viewing the program, at which time the system or method creates the video dynamically. To effectively use the transmission bandwidth, the system, instead of sending fully finished video frames and audio tracks, sends various pieces of raw content and instructions on how to create the final program from the raw contents. Since the system does not produce the final program until a viewer watches it, the system allows changes to the program at any time, even during playback. Using an Elastic Edit Decision List (EDL) (816), the system allows program customization including removing segments of shows already seen by the viewer, expanding segments in which the viewer previously expressed interest, and customizing a commercial content to the viewer's tastes. Allowing broadcasting high–quality television streams over low–bandwidth channels, the invention is advantageous over prior art techniques which transmit video and audio tracks in their entirety and are thus limited by the available bandwidth.

*FOR THE PURPOSES OF INFORMATION ONLY*

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **AL** | Albania | **ES** | Spain | **LS** | Lesotho | **SI** | Slovenia |
| **AM** | Armenia | **FI** | Finland | **LT** | Lithuania | **SK** | Slovakia |
| **AT** | Austria | **FR** | France | **LU** | Luxembourg | **SN** | Senegal |
| **AU** | Australia | **GA** | Gabon | **LV** | Latvia | **SZ** | Swaziland |
| **AZ** | Azerbaijan | **GB** | United Kingdom | **MC** | Monaco | **TD** | Chad |
| **BA** | Bosnia and Herzegovina | **GE** | Georgia | **MD** | Republic of Moldova | **TG** | Togo |
| **BB** | Barbados | **GH** | Ghana | **MG** | Madagascar | **TJ** | Tajikistan |
| **BE** | Belgium | **GN** | Guinea | **MK** | The former Yugoslav Republic of Macedonia | **TM** | Turkmenistan |
| **BF** | Burkina Faso | **GR** | Greece | | | **TR** | Turkey |
| **BG** | Bulgaria | **HU** | Hungary | **ML** | Mali | **TT** | Trinidad and Tobago |
| **BJ** | Benin | **IE** | Ireland | **MN** | Mongolia | **UA** | Ukraine |
| **BR** | Brazil | **IL** | Israel | **MR** | Mauritania | **UG** | Uganda |
| **BY** | Belarus | **IS** | Iceland | **MW** | Malawi | **US** | United States of America |
| **CA** | Canada | **IT** | Italy | **MX** | Mexico | **UZ** | Uzbekistan |
| **CF** | Central African Republic | **JP** | Japan | **NE** | Niger | **VN** | Viet Nam |
| **CG** | Congo | **KE** | Kenya | **NL** | Netherlands | **YU** | Yugoslavia |
| **CH** | Switzerland | **KG** | Kyrgyzstan | **NO** | Norway | **ZW** | Zimbabwe |
| **CI** | Côte d'Ivoire | **KP** | Democratic People's Republic of Korea | **NZ** | New Zealand | | |
| **CM** | Cameroon | | | **PL** | Poland | | |
| **CN** | China | **KR** | Republic of Korea | **PT** | Portugal | | |
| **CU** | Cuba | **KZ** | Kazakstan | **RO** | Romania | | |
| **CZ** | Czech Republic | **LC** | Saint Lucia | **RU** | Russian Federation | | |
| **DE** | Germany | **LI** | Liechtenstein | **SD** | Sudan | | |
| **DK** | Denmark | **LK** | Sri Lanka | **SE** | Sweden | | |
| **EE** | Estonia | **LR** | Liberia | **SG** | Singapore | | |

# CLIENT-SIDE DIGITAL TELEVISION AUTHORING SYSTEM

## BACKGROUND OF THE INVENTION

### 1. Field of the Invention

This invention relates generally to multimedia production, and more particularly to producing low bandwidth, high quality multimedia content programs in client-side systems.

### 2. Description of the Background Art

As computer and media technology improves, there is an increasing need for sending video over various digital distribution media, such as corporate intranets, by modem over telephone lines point-to-point, and over the Internet. Transmitting high quality video over these media traditionally requires high bandwidths, without which the final product is often erratic and of low resolution for viewing.

Current digital video technologies generally are associated with high data volumes and thus require high transfer rates. For example, the ITU-R 601 standard used in professional digital video production provides for an NTSC frame of 720 pixels x 486 scan lines, with eight-bit 4:2:2 sampling of Y, R-Y, and B-Y color components. At sixty fields per second, this frame structure results in a 20 megabyte per second data stream, which far exceeds rates currently available in common networks.

Presently, fast modems connected to the Internet transfer up to 56 kilobits per second. Transferring a high quality compressed (for instance, 5:1) ITU-R 601 signal would require a further compression of 750:1. Compression of this magnitude would render the underlying signal unrecognizable. Even a 30:1 compressed image stream,

which is relatively good quality, would require a further compression of 125:1. Various alternative techniques, such as smaller picture sizes and slower frame rates, enable the compressed video stream to be viewable, but these techniques degrade the images.

Digital networks have similar limited bandwidth deficiencies. Examples of digital networks include modem-to-modem over the conventional telephone network, the Internet with modems, the Internet with an Integrated Services Digital Network (ISDN) line at one end, the Internet with ISDN lines at both ends, and a corporate 10 Base T Local Area Network (LAN).

Latency presents additional problems for digital networks. The Internet architecture does not guarantee any particular transfer bandwidth to be continuous over time. Existing video streaming architectures attempt to solve this problem by filling a buffer in the client (receiver) side with future video and audio frames so that, when a gap occurs in the Internet transmission, video is played from the buffer with the hope that the transmission will resume before the buffer is emptied. However, the transmission often does not resume in time, which, as the frame rate drops, can cause a pause in the video stream accompanied by an audio glitch. The buffering techniques also require that the buffer be filled before the video program begins playing. Consequently, as the size of the buffer increases, the time delay before the video playback begins also increases. Additionally, avoiding glitches in playback requires that the buffer be re-filled before the next transmission interruption.

Current Internet multimedia is a poor substitute for media content which has been available on television for decades. The World Wide Web (the "web") is a collection of millions of electronic documents consisting primarily of text and still images, linked

together and accessible by anyone with a suitable Internet browser. This huge database of information is typically navigated, or "surfed," by using various search tools to find the desired information. Surfing, or navigating, the Internet is referred to as an "active" task in which the user actively engages in, and indeed controls, the exploration and acquisition of information.

In contrast, viewing traditional film and television programs is "passive," in that someone else (the storyteller, news playback engines, etc.) is conveying information to the viewer with no involvement on the viewer's part (other than perhaps changing the channel to a different program). Both active and passive experiences are desirable, and much activity has been focused on attempting to create a compelling passive experience over the Internet. For example, "push" technologies in newer Internet software packages allow users to subscribe to various "channels" offered by web sites. Multimedia-style presentations are then delivered ("pushed") for later viewing, without explicit requests for the data in the presentation. These multimedia animations currently consist largely of filled vector (outline) shapes and text, which animate on-screen. Although more interesting than static imagery, this animation technology is not as sophisticated as broadcast television graphics imagery, often moving over small areas of the screen, and at slow, non-uniform frame rates. Given that most consumers are used to watching high-quality video productions on television, passive multimedia animations will not achieve broad appeal in their current form.

In light of the prior art deficiencies, what is needed is a multimedia system that provides traditional broadcast-quality context while, at the same time, enabling efficient transmission over a low bandwidth communication system.

## SUMMARY OF THE INVENTION

The present invention provides a multimedia system and method for producing a client-side low-bandwidth television (LBTV) program. The invention first creates scripts of the programs in an authoring system and stores them on a server, which, upon receiving a request from a user to view a program, transfers the corresponding Elastic Edit Decision List (EDL) and a small amount of preliminary content to the user's client system. The EDL serves as the script and includes instructions for the client system to produce and orchestrate the program in real-time on the client's hardware.

The client system then executes the EDL for viewing the program with as little delay as possible. Even if a portion of the program script is received before all of the content has been transferred, the client system plays that portion of the script. While the client system is running one program segment, the program searches for data for subsequent segments, which may be locally available in the client system or be transferred from the server. If the data is not available, then the program may ask the client system to generate the data, use another piece of data having similar functions, or perform other functions to keep displaying the program continuously and smoothly, including prolonging display of an image, redisplaying an image that has been displayed earlier, or playing some graphic animation. Further, while the program is running, the server sends in the background additional raw content that is not available in the client system.

The invention, allowing producing the program and receiving raw program content in real-time, is advantageous over prior art techniques in which the final program including the video and audio tracks are transmitted in their entirety. The invention, to

effectively use transmission bandwidths, sends only a description of how to recreate a given scene, rather than sending every video frame. Consequently, the invention enables the use of low bandwidth media to broadcast high-quality imagery, at least as good as in a television broadcast.

## BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a schematic overview of a network comprising the authoring system and clients of the preferred embodiment of the invention;

FIG. 2 is a block diagram illustrating the architecture of the authoring system;

FIG. 3 is a schematic showing a timeline editor which is used to produce the event-driven storyboard for an LBTV program;

FIG. 4 shows an exemplary storyboard representing an LBTV program created in accordance with the invention;

FIG. 5 shows an LBTV program memory map created from the storyboard of FIG. 4;

FIG. 6 is a schematic overview of an architecture of a client system in FIG. 1;

FIG. 7 is a schematic showing a playback engine of a client system; and

FIG. 8 is a flowchart illustrating the method of the invention from the step of creating the LBTV scripts through the step of playing the LBTV program.

## DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Referring now to FIG. 1, a schematic overview is shown of a network 100 of the preferred embodiment, which includes an authoring system 104, a server 108, and a plurality of client systems ("clients") 110A to 110N, all of which interconnect via the Internet 114. Authoring system 104 enables generation of Low Bandwidth Television (LBTV) software program scripts, which include raw data and instructions to construct a "client-side" video/television program from raw data.

In the preferred embodiment, server 108 stores the scripts created by authoring system 104 and transmits the scripts to clients 110. Program scripts preferably reside as disk files on server 108. Other network distribution systems, such as corporate intranets, Local Area Networks (LANs), Wide Area Networks (WANs), 10 base T networks, and terrestrial digital Radio Frequency (RF) broadcasting, or modem (point-to-point) distribution systems are also effective.

When a viewer desires to watch a program, the viewer may cause the viewer's client 110 to contact server 108. Server 108 transfers the program script consisting of an Elastic Edit Decision List (EDL) file, along with a small amount of preliminary content to client 110. The Elastic EDL file serves as the script by which client 110 will orchestrate the final production of the video. Client 110 begins executing the EDL file with as little delay as possible. Additional raw content that is not already stored on client 110 is sent from the server 108 in temporal order over the Internet 114. Once the EDL file begins executing, the program sequentially displays the media content according to the script. Unlike many conventional computer media systems, however, program execution does not stop and wait for the availability of content. If a given section of the

program occurs before all of the content has been transferred, client 110 continues and plays that section of the program, typically using "stand-in" content, such as line drawings or substitute images. Elastic EDLs, developed by Play, Inc., a corporation having headquarters in Rancho Cordova, are an improvement over the traditional EDL that includes an inflexible list of video clip segments for producing the video in order of the segments. In contrast, an Elastic EDL is "elastic," that is, it contains information and instructions to create the LBTV program on the fly, including information on whether to adjust the program during displaying such as prolonging a video segment or substituting the segment with another segment, etc. Consequently, an Elastic EDL, when required to compensate for the bandwidth variations, allows scaling of the program. Further, an Elastic EDL, taking accounts a user's previous playing of a program, may contain a user's preferences, and allows subsequent playing based on these preferences.

Although FIG. 1 illustrates a preferred embodiment in which the authoring system 104 and client 110 are separated by the Internet 114, alternatively, a single application program may contain the authoring system 104. The application program is loaded by a user on client 110, which then can create, edit, and execute program scripts. In this alternative embodiment, server 108 may contain content which is utilized by the program scripts, or all necessary program content may be loaded onto client 110 as part of the application program. In addition, program scripts created by one client 110 can be executed by other clients 110 in such an embodiment.

Referring now to FIG. 2, a block diagram illustrates the architectural details of the authoring system 104. Authoring system 104 includes a data bus 201 connecting a Central Processing Unit (CPU) 204 to a plurality of memories, which store data and

application programs that run on CPU 204. These memories and applications include a timeline editor 208, a music sound track module 212, an audio effects module 216, a video content module 220, a video effects module 224, a text generation module 228, and a communication unit 236.

Timeline editor 208 is used to generate an event-driven storyboard for an LBTV program from music sound tracks, audio effects, video, video effects, and text stored in respective modules 212, 216, 220, 224, and 228. Timeline editor 208 can also retrieve audio and video objects from a computer hard disk or other means such as a scanner or a drawing tool. To reduce the video clip size, timeline editor 208 divides a long video clip into short pieces of video tracks, and when transferring the data to client 110, server 108 sends the entire audio tracks and these short pieces of video tracks. If all video tracks have not been received during display, client system 110, in accordance with instructions in the Elastic EDL, may replace the belated video tracks with cut-aways of camera-stand style still movies, animating graphics, or stock video clip footage.

Music sound track module 212 contains information associated with music. For example, music sound track 212 defines how long a piece of music will be played, how loud the musical piece will be played, what music key the piece will be performed in, and whether the music will be looped for continuous playing. The music is preferably stored as MIDI or compressed .WAV files.

Audio effects module 216 produces various audio effects, such as control of music tracks, Foley effects, and voice-overs. Audio effects module 216 can also mix multiple audio tracks and add special effects, including reverberation and delays in real-time.

Video content module 220 contains the video objects from which an LBTV program is created. These video objects are stored in a variety of forms such as still pictures, short video clips, animated computer graphics, bitmaps, or other movie files suitable for QuickTime movies. Video content module 220 also contains information about how the video objects are to be used, e.g., whether a picture will be displayed in full size or half size, and whether the picture will be cropped or trimmed to exclude unwanted images.

Video effects module 224 enables application of a variety of visual effects to picture images in an LBTV program. Examples of video effects include cuts, dissolves, fades to and from black, wipes, organic animating wipes, and digital video effects such as push, pull, and flying video planes. Additionally, video effects module 224 can emulate various conventional camera techniques used in traditional video production, such as slow camera pans and zooms across still images. Video effects module 224 also provides transitional smoothing from one scene to another, from one image still to another, and from one video clip to another.

Text module 228 may be a conventional text processing program and is used to design and generate text to be displayed with the video.. Preferably, text module 228 enables text to be created as an object and placed anywhere on a video page.

Authoring system 104 uses communication unit 236 to transmit and receive data from other systems, including server 108 and clients 110. Communication unit 236 is a conventional network communication device such as a router or modem.

Referring now to FIG. 3, a block diagram illustrates the architecture of the timeline editor 208 that is used to produce the video storyboard for an LBTV program.

Timeline editor 208 includes a source element browser 308, an element library 312, a plurality of object property editors 320, a storyboard display 326, an LBTV script compiler 330, and a playback engine 334. Source element browser 308 enables a user to find video objects and bring them into timeline editor 208. These video objects may reside on an Internet web page, a CD-ROM, or a PC hard disk, but preferably are contained in the element library 312 connected to element browser 308. These video objects may also be provided from a scanner, a software video object drawing tool, or other means of providing video elements. Additionally, the video objects can be "dragged" and "dropped" into timeline editor 208 for editing.

Each type of video object is edited using a specific object editor 320 suitable for the particular type of object. For example, text objects are conventionally edited using a word processor, still pictures are generally edited using a "Paint" program, and so on. Although object editors 320 are shown as residing within timeline editor 208, object editors 320 could reside anywhere in authoring system 104, so long as they can be accessed by timeline editor 208. Video objects are displayed, preferably after having been edited, in a storyboard format using the storyboard display 304. A storyboard representation of an LBTV program is preferably displayed as a sequence of icons or pictures. Each icon in the sequence represents a script object. For example, a still image in the sequence might be represented by a thumbnail of the image. A music object (.WAV file) might be represented by a music symbol displayed with the name of the file. An animated sequence might be represented by a thumbnail image of the first frame of the sequence. LBTV compiler 330 generates an LBTV program script from the edited

storyboard. Playback engine 334 allows a user to review a program script created by timeline editor 208.

Referring now to FIG. 4, a block diagram illustrates an exemplary storyboard 400 of an LBTV program created in accordance with the invention. Storyboard 400 is event-driven and includes, for example, from time $T_0$ to time $T_5$, a "fade from black," a still image A, video effects for the still image A, a still image B, a virtual camera pan, and a still image C, respectively.

Initially at time $T_0$ the program preferably starts by "fading from black" to still image A at $T_1$. Other colors and/or effects may substitute for the fade-from-black transition. The fading time is preferably synchronized to a real-time clock, and is independent of the hardware and/or graphic card of the client 110. At time $T_1$ still image A is displayed. At time $T_2$ video effects are added to enhance image A. Examples of the video effects applied to still image A include wipes, organic animating wipes, and digital video effects such as push, pull, and flying video planes. Other audio and musical effects may also be added. At time $T_3$, still image B is displayed, and like still image A, image B may be accompanied by video and audio effects. At time $T_4$, a virtual camera pan is used to pan across image B. The virtual camera pan is a video effect in which the viewer is able to see only a portion of the still image being viewed. The portion of the image displayed to the viewer changes slowly, giving the viewer the perspective of either moving his eyes across a panoramic view or of having the image move relative to the viewer. The virtual camera pan effect at $T_4$ contains video data related to the panning paths so that the paths can be constructed in client 110. At time $T_5$, image B is replaced by still image C. Since no transition video effect separates still images B and C (such as a fade or a wipe), image C merely replaces image B.

As the FIG. 4 storyboard display 304 executes, each subsequent image, sound, or video effect is loaded by playback engine 334 for execution. For a variety of reasons including a large file size, network (such as Internet 114) latency, or resource congestion, the image or sound may not be immediately available for playback. When images are unavailable, programmed playback engine 334 continues executing the storyboard by substituting some alternative content for the missing image. For example, if at time $T_3$ image B is not available to replace the still image A of time $T_1$, the program may continue to display still image A. Alternatively, a line drawing or stock photograph may be substituted for the image A. Another alternative can be that playback engine 324 loops to groups of images that have been displayed earlier. In the storyboard 400 the invention continuously redisplays events from time $T_1$ to time $T_3$ that include still image A, video effects for image A, still image B, and virtual camera panning for image B. Other options to keep the program running continuously include looping the music, looping the graphics animation, and replaying parts of the storyboard. The invention can also substitute stand-in video content, change picture color to black and white, or substitute the transitions from one image to another image. In any event, as with traditional television, the programming must continue to play. This is in contrast to more traditional multimedia systems where the playback system generally stops executing and waits for the missing image video or audio clip to load before playback resumes.

Referring now to FIG. 5, a block diagram illustrates an LBTV program memory map 500 containing the program script created from the storyboard 400 of FIG. 4. In the program script, a set of parameters is associated with each FIG. 4 time event. These parameters are stored in memory sections 504, 508, 510, 512, 516, and 520 for events at time $T_0$, $T_1$, $T_2$, $T_3$, $T_4$, and $T_5$, respectively. In section 504, the fade from black event

includes black as the initial color, and one second as the minimum duration time for fading. In section 508, image A is stored as a compressed bitmap. Additional parameters appended to image A indicate that image A will be displayed for a minimum of five seconds, and has a cropping value of ($X_1$, $Y_1$, $X_2$, $Y_2$). Image A also has a zoom factor Z. Although in the embodiment shown in FIG. 5, the images (including still image A) are shown stored in sequential memory, memory section 508 in fact preferably points to an image library where image A is actually stored.

Section 510 contains the video effect occurring at time $T_2$. The video effect is described as having a video effects speed of one, with still image A traveling from left (L) to right (R), and image A having a red border.

Section 512 contains information for compressed image B. Image B is described as being displayed for ten seconds. Following this ten second display of image B, a virtual camera pan is described in section 516 as occurring with respect to image B. As previously described, the virtual camera pan is a pre-programmed effect, which in this case preferably results in a left to right sweep of the image across the video screen. The virtual camera pan of section 516 indicates that no zoom of the still image B will occur. Memory section 520 stores the still image C described with reference to FIG. 4 as being displayed at time $T_5$. In the program script corresponding to the storyboard 400 of FIG. 4, still image C is the last element.

Referring now to FIG. 6, a block diagram illustrates the preferred architecture of FIG.1 client 110, which can be a standard personal computer (PC), an LBTV compatible set-top box, or other system capable of executing software and displaying multimedia content. Since LBTV programs are actually created in a high-level description language,

the program's actual content can be generated in real-time to fit the capabilities of the client 110 system. In the case of a PC, client 110 is preferably a Pentium computer with a PCI video display card. The real-time playback of an LBTV program preferably does not require a graphics card with any special hardware capabilities. In the preferred embodiment, client 110 works well on a PC, without compromising the broadcast television mandates of smooth, continuous, and well-produced content. LBTV-compatible set-top boxes also preferably use standard PC display technologies, resulting in a low cost, stand-alone unit. In both PC and set-top configurations, the LBTV architecture preferably combines the standard PC display chipsets, a high-speed, general purpose CPU, and flexible software application modules to maintain a professional quality television program, even over a low bandwidth connection.

Client 110 includes a client CPU 604 connected via data bus 601 to a plurality of memories which store data and application programs that run on the CPU 604, including a network browser 608, a playback engine 612, a script library 616, a video library 620, an algorithmic video and audio module 628, an audio library 634, and a keying and layering engine 638. A display 640 and a communication unit 642 are also conventionally coupled to data bus 601.

Internet browser 608 allows a user to request a stored LBTV program, such as from a web site, and to download various media resources (images, effects, video clips, etc.) that might be required. Playback engine 612 provides intelligence for an LBTV script to be executed for viewing. In the preferred embodiment, playback engine 612 of FIG. 6 is identical to playback engine 334 of FIG. 3 discussed with reference to timeline editor 208 of authoring system 104. In one alternative embodiment, playback engine 334

has a different user interface or diagnostics than the client playback engine 612, as might be contemplated for an editor versus a run-time tool. Script library 616 stores the program scripts that have been created and transferred to client 110. Each program script includes the Elastic EDL and some small preliminary content. Since the invention does not produce a client-side LBTV program until a viewer watches it, the program may be modified at any time, including during playback. The invention, using instructions in the Elastic EDL, may, in real-time, customize a program for the viewer, including removing segments of shows already seen by the viewer, expanding segments which include topics the viewer previously expressed interest in, and customizing a commercial to the viewer's tastes. The invention may also produce, in real-time, video edits, special effects, titles, graphics, and audio streams.

When a viewer starts playing a program, using a script stored in the library 616, additional raw content that is not on client 110 is sent from server 108 in temporal order. Audio and video objects, normally transferred from server 108 to create the LBTV program, are stored in audio library 634 and video library 620, respectively. Pre-generated canned video elements such as borders, shadows, etc., are also stored in video library 620. When a program seeks to display an audio or video object, the program searches for that object in the appropriate library 634 or 620. Algorithmic video and audio module 628 is used to generate video images based on a defined algorithm provided by playback engine 612. Keying and layering 638 provides the capabilities for a viewer to see one picture layered on another picture, or one video object superimposed over a background. Client 110 uses communication unit 642 to transmit and receive data from server 108.

Referring now to FIG. 7, a block diagram illustrates the playback engine 612 which orchestrates the final production of an LBTV program. Playback engine 612 includes a speed coordinator 704, a synchronization module 708, and a transitional smoothness module 712. Speed coordinator 704 is responsible for ensuring that the LBTV program scripts run equivalently on both slow (PC 486) and faster (PC Pentium) machines. A program that runs on a faster machine does not necessarily complete its tasks in a shorter time, but the program's audio and video objects are usually of better quality. Further, a program created for a faster machine, using a mapping technique, can run on a slower machine. For example, when running on faster machines a program uses a 24-bit color representation for higher-quality video, but will map to an 8-bit color representation when running on slower machines. Similarly, audio in slower machines can also be mapped using fewer bits than the normal 16-bit representation for faster machines. Speed coordinator 704 first determines the speed at which playback engine 612 is running, and, based on this speed, takes actions accordingly. The speed of playback engine 612 varies, depending on the speed of other system components, such as graphic card, audio card, system mother board, etc.

Synchronization module 708 establishes synchronization points to synchronize different audio and video tracks that form the LBTV program. In between each two synchronization points the tracks may not synchronize, but at each synchronization point the tracks re-synchronize. In the preferred embodiment, one synchronization point is close enough to an adjacent point so that a program viewer cannot perceive that the tracks do not synchronize. Additionally, at each synchronization point, module 708 waits to

acquire all the data required for the next program segment before allowing the segment to be displayed.

Transitional smoothness module 712 provides the high-quality, smooth and continuous appearance of a conventional television program. While the program is waiting to acquire all necessary data, module 712 uses several techniques to "mask out" the wait state so that a viewer perceives a smooth and continuous program. Module 712 can instruct the program to fade out, prolong playing on one video/audio track, loop on one track continuously, or replay part of a track. Module 712 can also use an animation technique allowing some video objects to move around display 640 while the program is waiting. This is done using the capabilities provided by the keying and layering module 638. Substitution can also be used. For example, if a program cannot play a desired musical instrument because the instrument is not available on the client 110 at the time needed, transition smoothness module 712 can replace the desired instrument with another similar instrument that is available on client 110. Similarly, if module 712 requires a video that is not available, module 712 can request that algorithmic video module 628 create a video replacement based on algorithms defined by module 712.

Referring now to FIG. 8, a flowchart illustrates the operation of the invention from the step of creating an LBTV program script through the step of playing the program. The method begins with creating the LBTV scripts in step 804. An LBTV program producer uses the authoring system 104 to generate the program scripts. Within the authoring system 104, timeline editor 208 is used to edit and add musical, audio, and video effects. A storyboard is created in which objects are sequentially arranged to produce the program script. Objects, represented by graphical icons within the storyboard, can be manipulated by dragging and rearranging the icons. The objects can

be moved to and from object libraries (e.g. music soundtrack 212, video content 220, video effects 224 (FIG. 2)) and downloaded from remote servers 108. A script compiler 330 converts the storyboard to an executable program file.

As described for the model shown in FIG. 1, various LBTV scripts may be created and in step 808 are transferred to server 108. Server 108 preferably sets up a web page and makes the scripts available to a viewer having access to the Internet 114. The web page of server 108 also preferably includes instructions for a viewer to request and download a desired LBTV program. Each LBTV program script can also be represented as a picture icon on the web page.

In step 812 a viewer uses an Internet browser 608 to request a program for viewing. In the preferred embodiment, the viewer contacts the server 108, which then in step 816 transfers the Elastic EDL along with a small amount of preliminary content. The EDL serves as the script to which the client system 110 orchestrates the final production of the LBTV video on the client 110's hardware. The invention, to effectively use the transmission bandwidth, sends only a description of how to recreate a given scene, rather than sending every video frame.

In step 820 client system 110 starts executing the Elastic EDL for viewing the program with as little delay as possible. The Elastic EDL contains information on whether there are sufficient content segments to start playing the program. The delay depends on the system bandwidth and speed. Even if a given section of the program script occurs before all of the content has been transferred, client 110 continues and plays that section of the script, typically using "stand-in" content, such as line drawings, stock photographs, etc. While client 110 is running one program segment, the program in step 821 is searching for data for subsequent segments. The data may be transferred from

server 108, or be locally available in the client 110's storage disk including video library 620 or audio library 634. Additionally, in step 822, while the program is running, server 108 in the background sends additional raw content that is not available in client 110.

If in step 824 the program finds data available, the data is retrieved in step 826 and the program continues to step 828. If the data is not available in step 824, the program has several options. One option is that the program can substitute another piece of data having a similar function. For example, if a section of audio requires a synthesized piano voice and the piano voice is not immediately available, a pipe organ or similar available instrument may be substituted. As such, step 836 determines whether the substitute data is acceptable or feasible. When substitution is feasible, the program in step 840 retrieves and provides the substitute data for the program to continue in step 828.

However, if in step 836 substitution is not feasible, the program in step 838 requests algorithmic video and audio module 628 to generate data suitable for the next program segment. In step 844, the program determines whether data from module 628 is available. If so, the program in step 848 then retrieves the data from module 628 and continues in step 828. If the program, pursuant to step 844, determines that such data is not available, the program performs other functions in step 852. These functions include, for example, prolonging display of an image, redisplaying an image that has been displayed earlier, or playing some graphics animation until the desired data is completely transferred from server 108. The program in step 828 continues to fetch data and run until the program is complete.

The invention has been explained above with reference to a preferred embodiment. Other embodiments will be apparent to those skilled in the art after reading

this disclosure. Therefore, these and other variations upon the preferred embodiment are intended to be covered by the appended claims.

WHAT IS CLAIMED IS:

1. A method for producing a multimedia program for viewing in a client system, comprising the steps of:

storing on a server a plurality of scripts each corresponding to a respective one of said programs;

transferring the script corresponding to one said program to said client system;

viewing said transferred program at said client system;

sending additional data in background to said client system while said program is being viewed;

retrieving said additional data by said client system; and

incorporating said additional data into said program as said program is being viewed at said client system.

2. The method of claim 1 wherein said program is substantially equivalent to a program selected from a group consisting of a television program, a QuickTime movie program, and a high-resolution imagery program.

3. The method of claim 1 wherein the step of sending comprises the steps of:

providing a video clip including a plurality of video tracks;.

providing an audio clip corresponding to said video clip and including an audio track;

sending said audio track to said client system; and

sending said video tracks in a piecemeal manner to said client system.

4. The method of claim 1 further comprising the step of sending instructions on how to create said transferred program.

5. The method of claim 4 wherein said instructions are in the form of an Elastic Edit Decision List.

6. The method of claim 1 further comprising the step of, after the step of transferring, producing said program in real-time.

7. The method of claim 6 further comprising the step of modifying said program in real-time.

8. The method of claim 7 wherein the step of modifying is selected from a group consisting of the steps of removing a segment of said program, expanding a segment of said program, and customizing contents of said program.

9. The method of claim 6 comprising the further step of, while a program segment of said program is being viewed, searching for data for subsequent segments.

10. The method of claim 9 wherein said data for subsequent segments is selected from a group consisting of data in said server, data in said client system, data generated in real-time, data having a similar function, animation data, and data that has been displayed earlier.

11. The method of claim 1 further comprising the step of sending a description of how to create a scene in said transferred program.

12. The method of claim 1 further comprising the step of using a communication connection selected from a group consisting of a low-bandwidth connection, a point-to-point modem, a cable modem, the Internet, a local area network, a wide are network, an intranet, and an terrestrial digital radio frequency band.

13. The method of claim 1 wherein said scripts constitute an event-driven storyboard.

14. The method of claim 1 comprising the further step of requesting for viewing said program.

15. A client-side multimedia-program authoring system, comprising:

means for storing an authoring system script on a server;

means for transferring said script to a client;

means for viewing said program at said client;

means for sending additional data in the background while said program is being viewed;

means for retrieving said additional data; and

means for incorporating said additional data into said program as said program is being viewed.

16. A computer-readable medium storing codes to cause a computer to produce a multimedia program for viewing in a client system, by the steps of:

storing on a server a plurality of scripts each corresponding to a respective one of said programs;

transferring the script corresponding to one said program to said client system;

viewing said transferred program at said client system;

sending additional data in background to said client system while said program is being viewed;

retrieving said additional data by said client system; and

incorporating said additional data into said program as said program is being viewed at said client system.

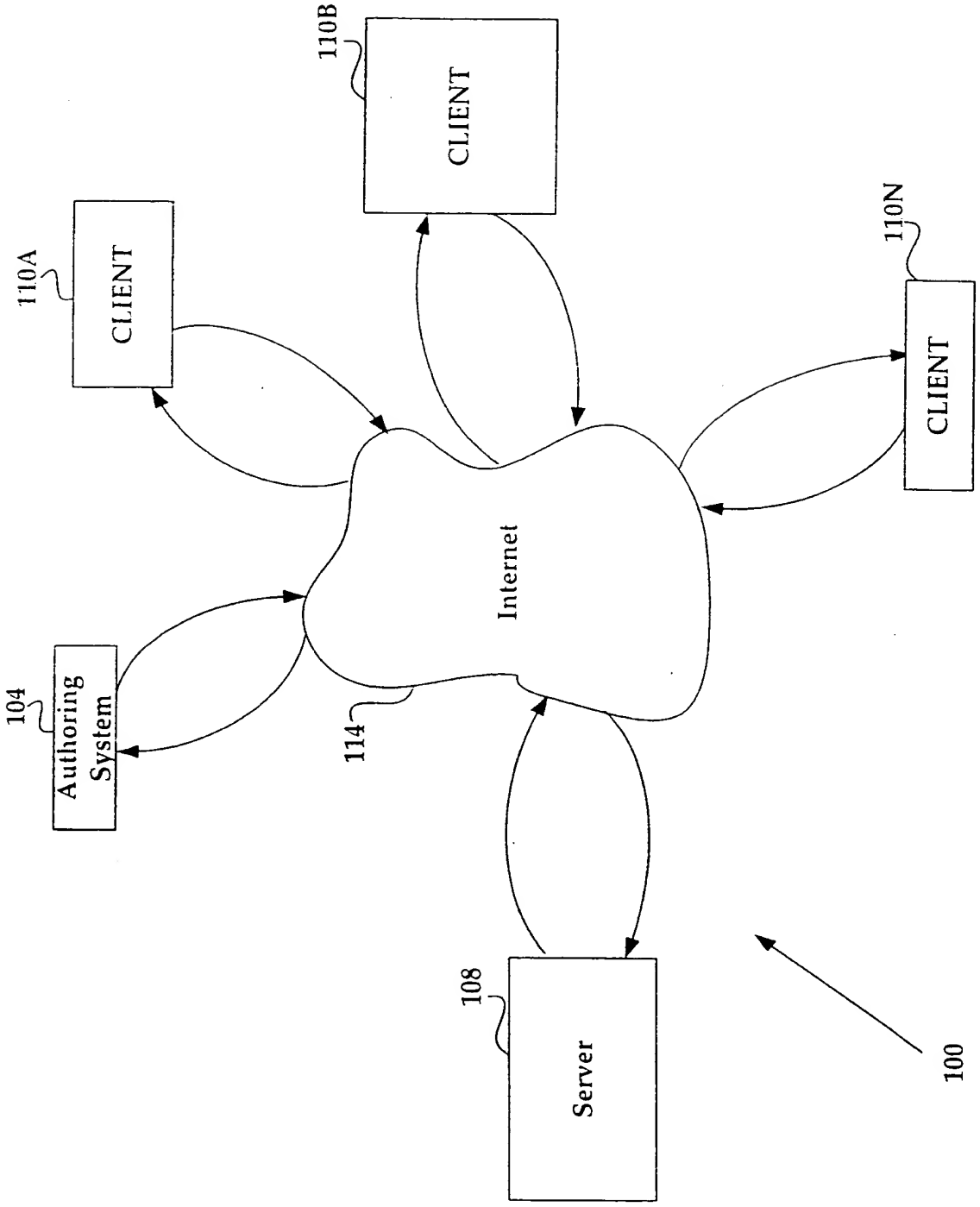

110B
CLIENT
110A
CLIENT
110N
CLIENT
Internet
104
Authoring System
114
108
Server
100

FIG. 1

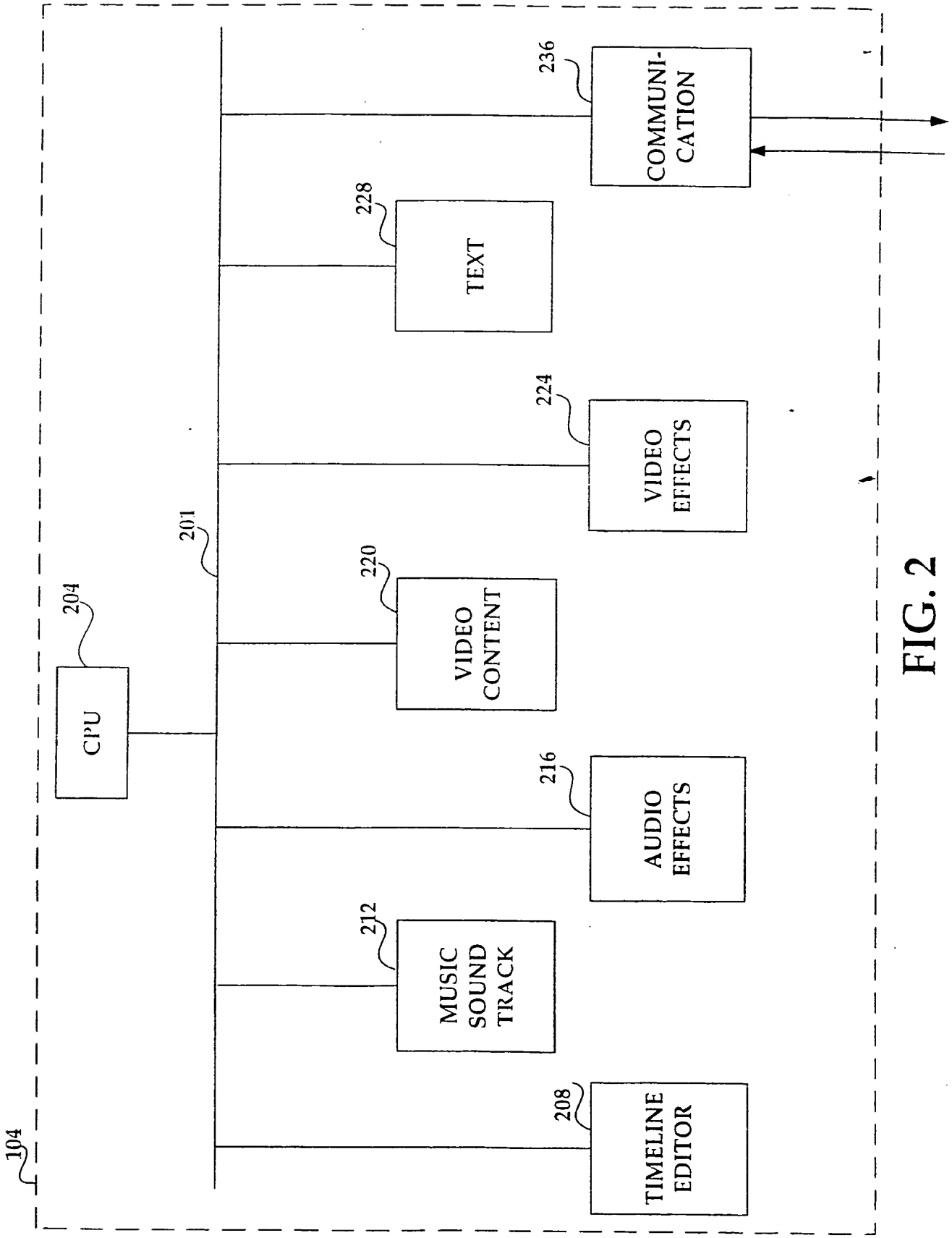

104
CPU
204
201
TIMELINE EDITOR
208
MUSIC SOUND TRACK
212
AUDIO EFFECTS
216
VIDEO CONTENT
220
VIDEO EFFECTS
224
TEXT
228
COMMUNI-CATION
236

FIG. 2

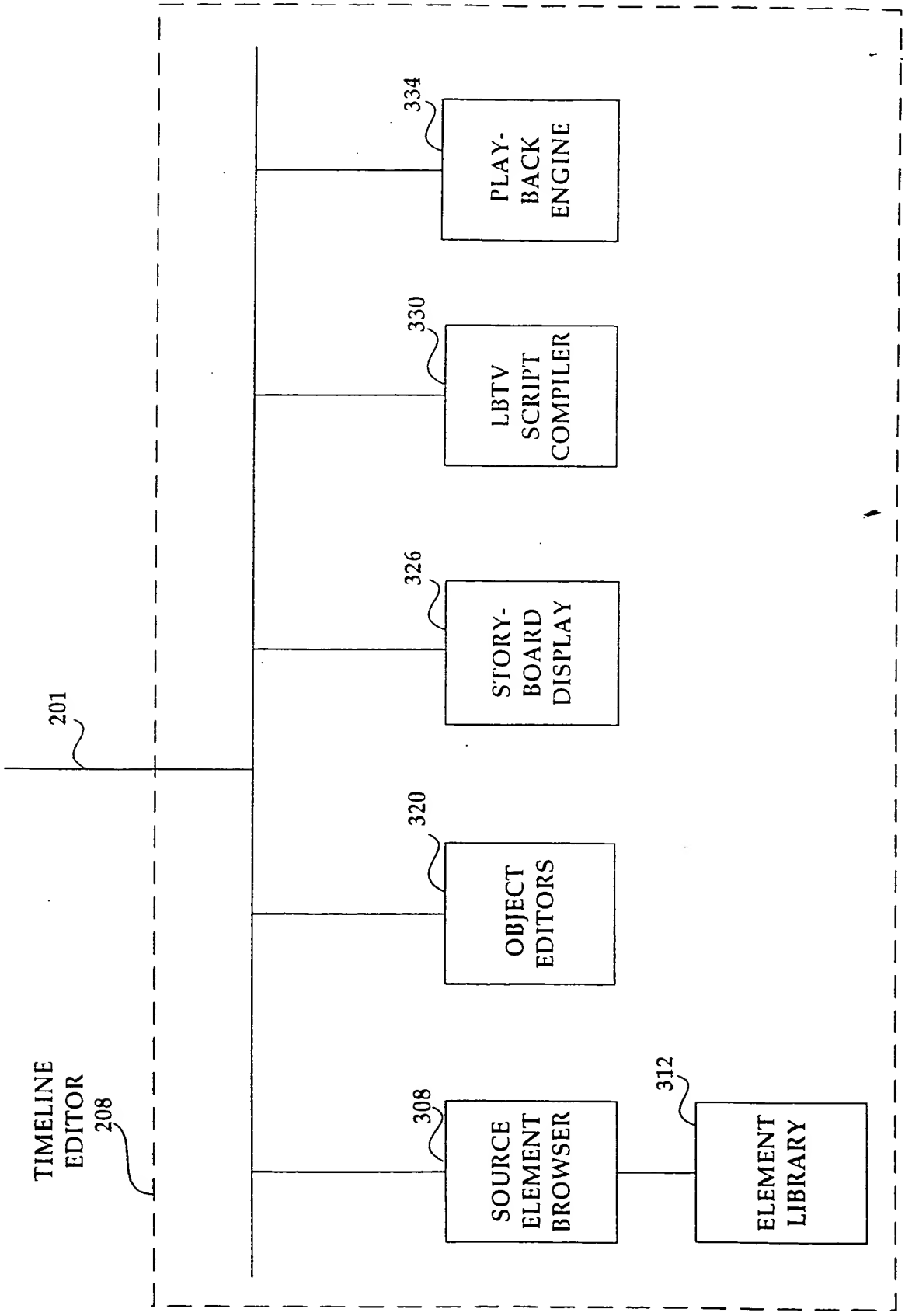

TIMELINE EDITOR
208
201
SOURCE ELEMENT BROWSER
308
ELEMENT LIBRARY
312
OBJECT EDITORS
320
STORY-BOARD DISPLAY
326
LBTV SCRIPT COMPILER
330
PLAY-BACK ENGINE
334

FIG. 3

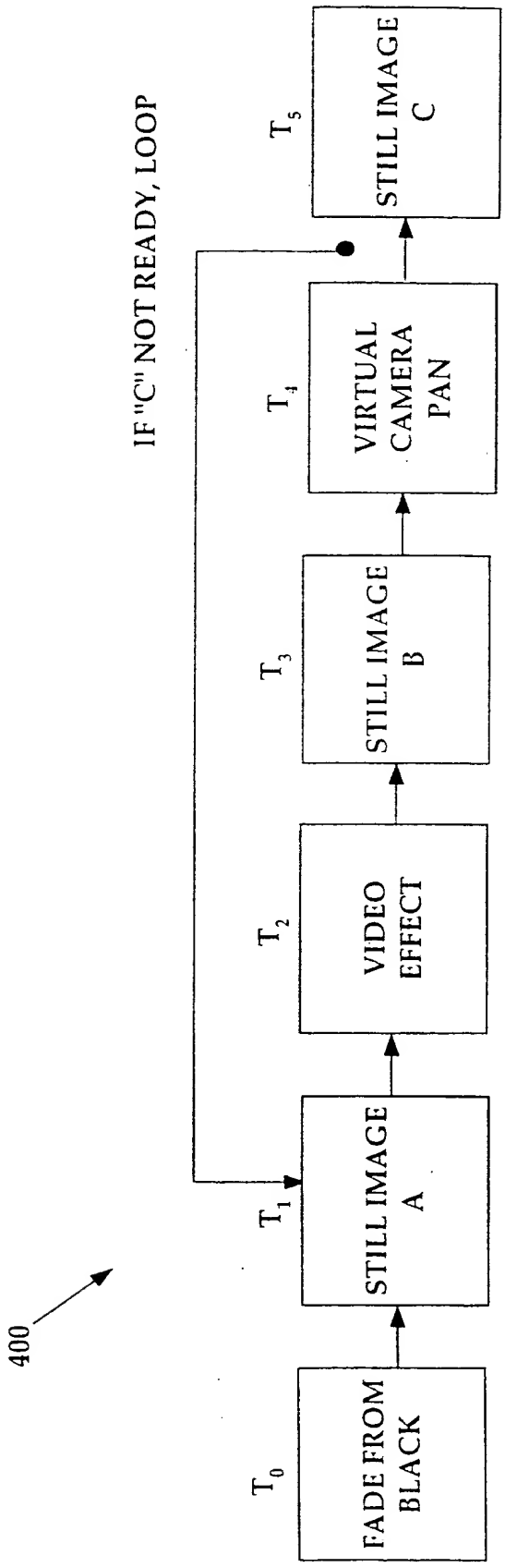

400
$T_0$
FADE FROM BLACK
$T_1$
STILL IMAGE A
$T_2$
VIDEO EFFECT
$T_3$
STILL IMAGE B
$T_4$
VIRTUAL CAMERA PAN
$T_5$
STILL IMAGE C
IF "C" NOT READY, LOOP

FIG. 4

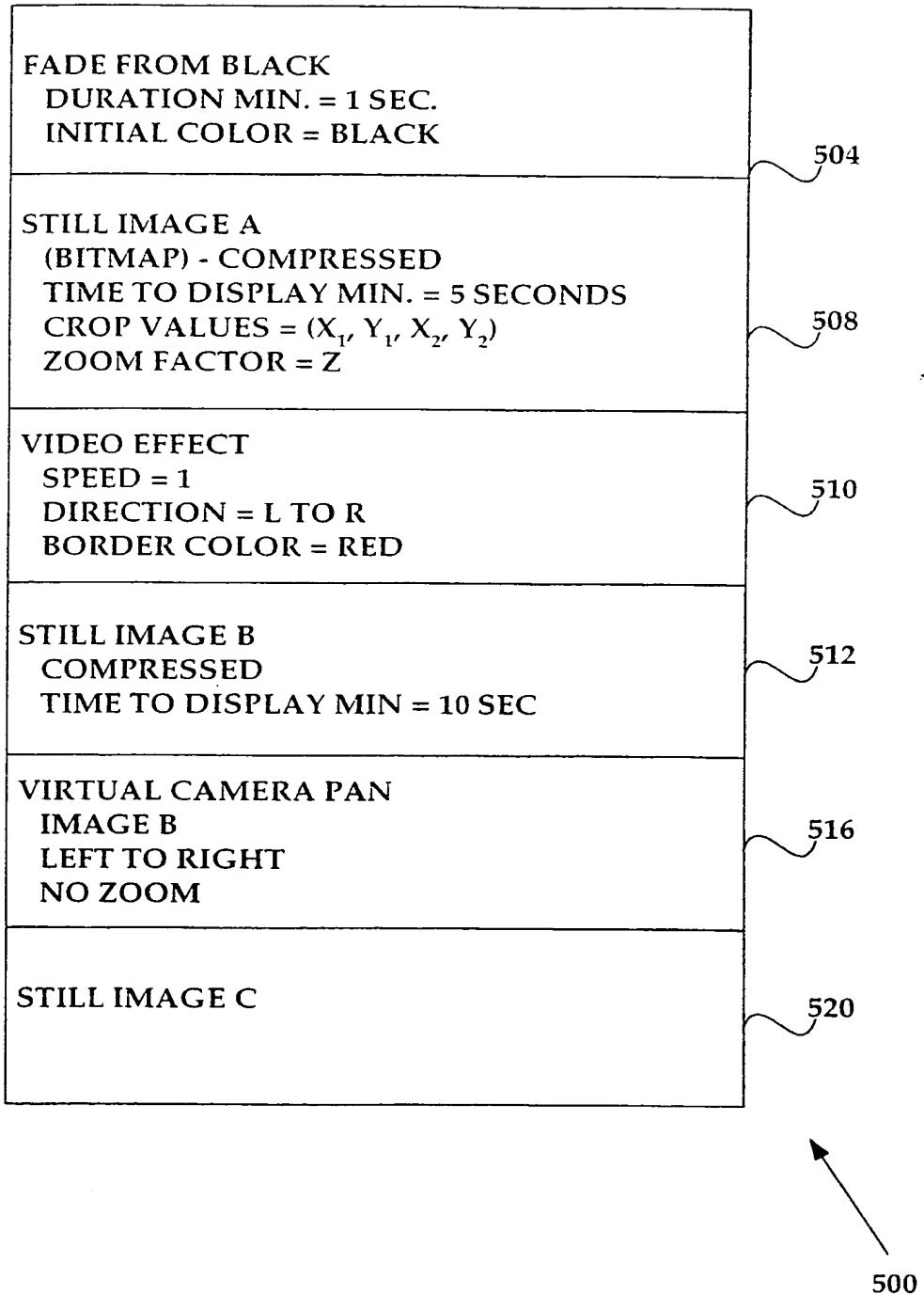

FADE FROM BLACK
DURATION MIN. = 1 SEC.
INITIAL COLOR = BLACK
504
STILL IMAGE A
(BITMAP) - COMPRESSED
TIME TO DISPLAY MIN. = 5 SECONDS
CROP VALUES = (X1, Y1, X2, Y2)
ZOOM FACTOR = Z
508
VIDEO EFFECT
SPEED = 1
DIRECTION = L TO R
BORDER COLOR = RED
510
STILL IMAGE B
COMPRESSED
TIME TO DISPLAY MIN = 10 SEC
512
VIRTUAL CAMERA PAN
IMAGE B
LEFT TO RIGHT
NO ZOOM
516
STILL IMAGE C
520
500

FIG. 5

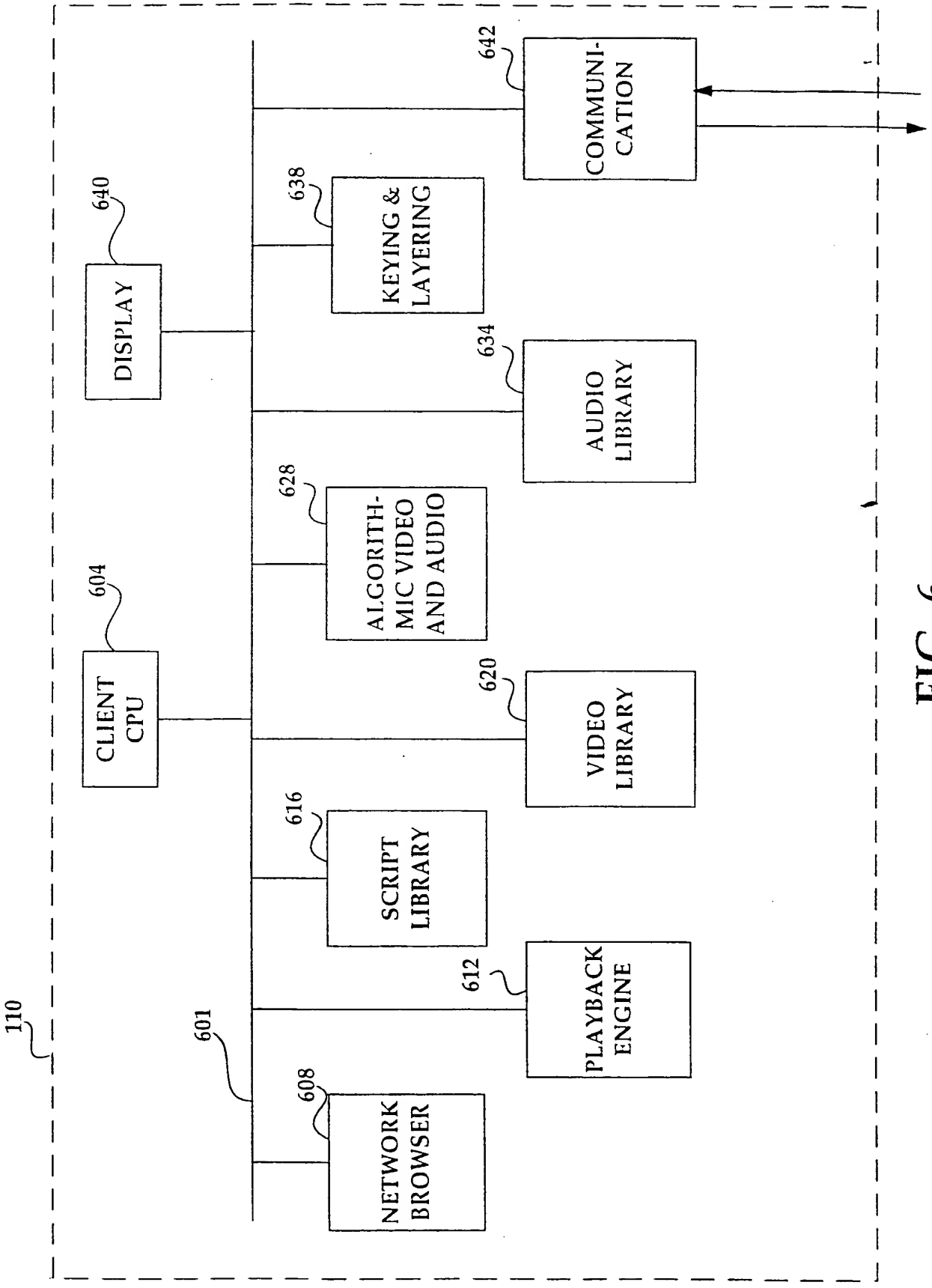

110
601
642
COMMUNI-
CATION
638
KEYING &
LAYERING
640
DISPLAY
634
AUDIO
LIBRARY
628
ALGORITH-
MIC VIDEO
AND AUDIO
604
CLIENT
CPU
620
VIDEO
LIBRARY
616
SCRIPT
LIBRARY
612
PLAYBACK
ENGINE
608
NETWORK
BROWSER

FIG. 6

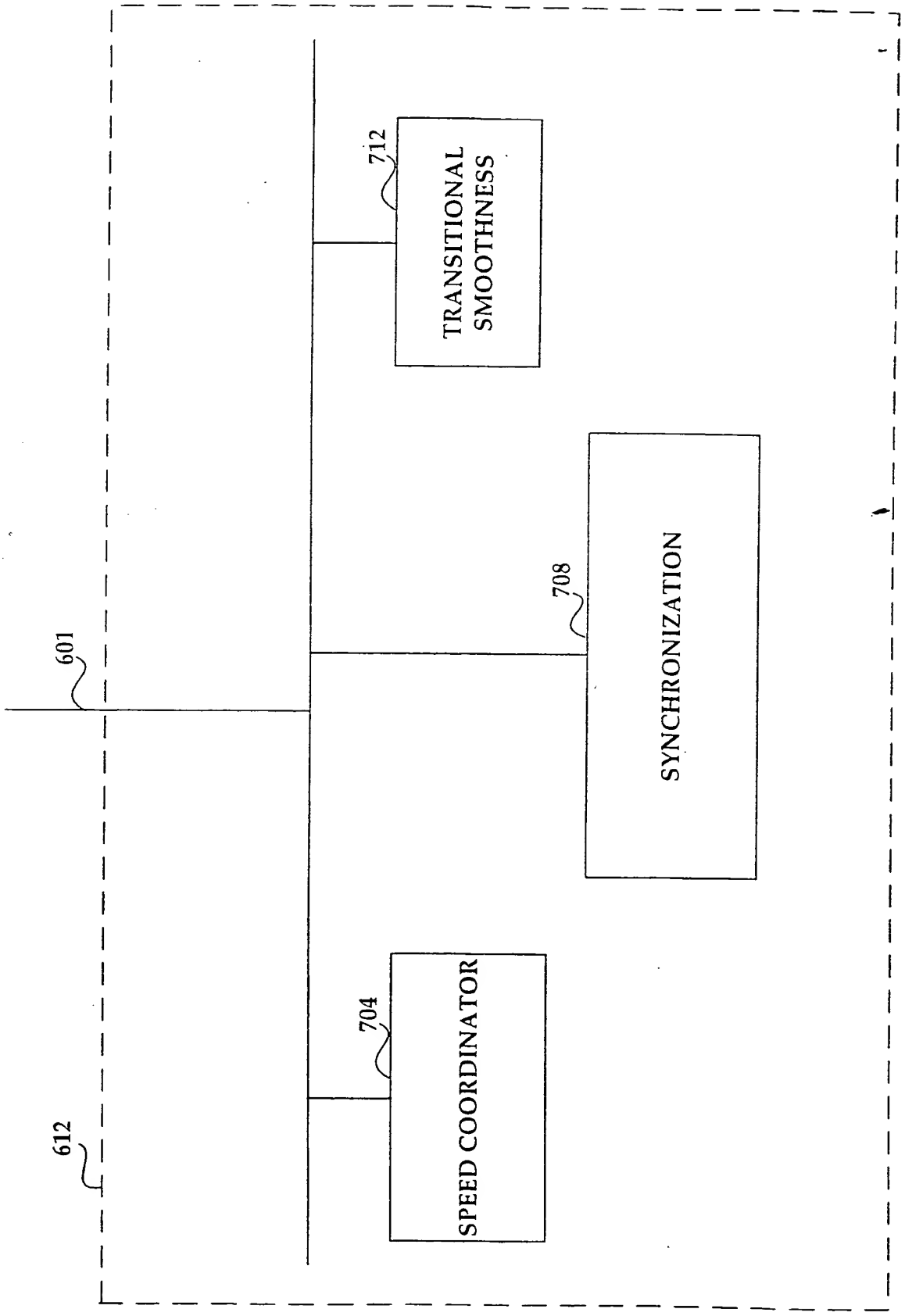

612
601
712
TRANSITIONAL SMOOTHNESS
708
SYNCHRONIZATION
704
SPEED COORDINATOR

FIG. 7

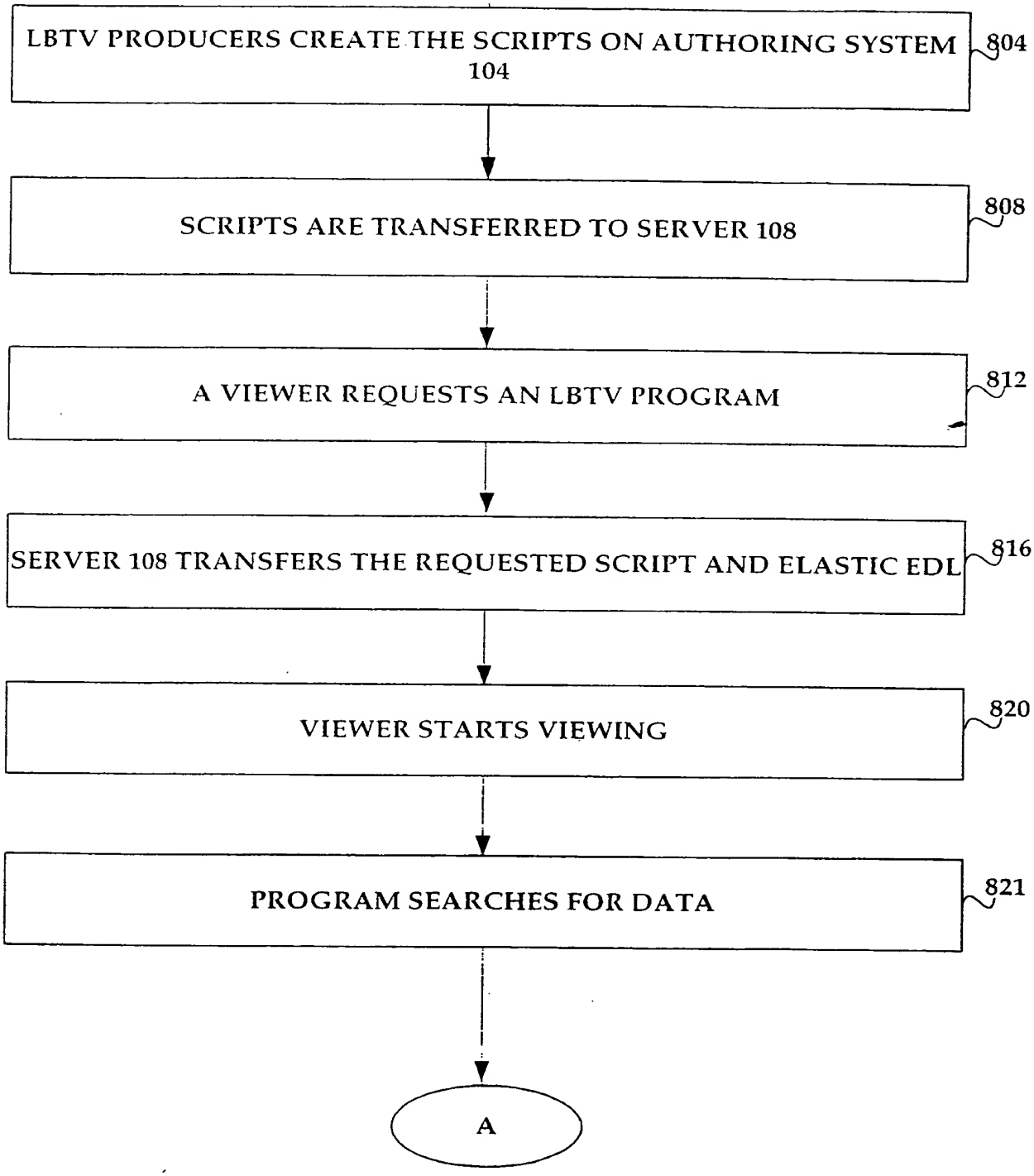

LBTV PRODUCERS CREATE THE SCRIPTS ON AUTHORING SYSTEM 104
804
SCRIPTS ARE TRANSFERRED TO SERVER 108
808
A VIEWER REQUESTS AN LBTV PROGRAM
812
SERVER 108 TRANSFERS THE REQUESTED SCRIPT AND ELASTIC EDL
816
VIEWER STARTS VIEWING
820
PROGRAM SEARCHES FOR DATA
821
A

FIG. 8